Sidebar

Sidebar The Sidebar component is a comprehensive navigation component designed to provide structured navigation, user management, and content organization in modern web applications. It features collapsible functionality, nested menu structures, user profiles, and integrated search capabilities. The <aava-sidebar> component offers a flexible and responsive sidebar solution that can adapt to different screen sizes and user preferences, making it ideal for dashboards, admin panels, and complex navigation systems. How to use ■ import { AavaSidebarComponent , MenuItem , UserProfile } from "@aava/play-core" ; Basic Usage ■ Simple sidebar with basic navigation and user profile. Angular Preview Code < div class = " sidebar-demo " > < aava-sidebar [width] = " sidebarWidth() " [collapsedWidth] = " collapsedSidebarWidth() " [showCollapseButton] = " true " [buttonVariant] = " ' inside' " [isCollapsed] = " isCollapsed() " [position] = " ' left' " (collapseToggle) = " onCollapseToggle($event) " > < div class = " demo-header-content " > < span class = " header-title " > My App </ span > </ div > < ul slot = " content " class = " demo-content " > < li *ngFor = " let item of menuItems " [class.active] = " item.active " class = " nav-item " > < span class = " nav-icon " > {{ item.icon }} </ span > < span *ngIf = " !isCollapsed() " > {{ item.label }} </ span > </ li > </ ul > </ aava-sidebar > </ div > isCollapsed = signal < boolean > (false) ; sidebarWidth = signal < string > ('280px') ; collapsedSidebarWidth = signal < string > ('70px') ; menuItems = [{ icon : '■' , label : 'Dashboard' , active : true } , { icon : '■' , label : 'Users' } , { icon : '■' , label : 'Projects' } , { icon : '■' , label : 'Analytics' } , { icon : '■■' , label : 'Settings' } ,] ; onCollapseToggle (collapsed : boolean) : void { this . isCollapsed . set (collapsed) ; } Features ■ Responsive Design ■ Automatic size adjustments based on screen dimensions Collapsible functionality for space optimization Mobile-friendly collapsed state Flexible Layout ■ Three size variants for different use cases Configurable header and footer sections Custom content projection slots Advanced Navigation ■ Nested sub-menu support with expandable sections Active state management for navigation items Icon support for visual navigation User Experience ■ Integrated search functionality User profile display and interaction Smooth collapse/expand animations Customization ■ Logo and branding options Theme-aware styling CSS custom properties for theming API Reference ■ Input Properties ■ Property Type Default Description width string '' Sidebar width collapsedWidth string '108px' Collapsed sidebar width height string '' Sidebar height hoverAreaWidth string '10px' Hover area width showCollapseButton boolean true Show collapse/expand button buttonVariant 'inside' | 'outside' 'inside' Collapse button position isCollapsed boolean false Initial collapsed state position 'left' | 'right' 'left' Sidebar position toggleOffset number 400 Toggle button offset togglePosition 'top' | 'center' | 'bottom' 'top' Toggle button position customStyles Record<string, string> {} Custom CSS styles Output Events ■ Event Type Description collapseToggle EventEmitter<boolean> Emitted when sidebar collapse state changes Content Projection Slots ■ Selector Description [slot=content] Custom content above the navigation menu [slot=footer] Custom footer content above user profile Best Practices ■ Menu Organization ■ Logical Grouping : Group related menu items together Hierarchical Structure : Use sub-menus for complex navigation Consistent Icons : Use meaningful and consistent icons for menu items Clear Labels : Provide descriptive text for navigation items Responsive Considerations ■ Mobile-First : Design for mobile devices first Collapsed State : Ensure collapsed sidebar remains functional Touch Targets : Maintain adequate touch target sizes Content Priority : Prioritize essential navigation in collapsed state User Experience ■ Search Integration : Implement search for applications with many menu items Active States : Clearly indicate current navigation location Smooth Transitions : Use smooth animations for state changes Accessibility : Ensure keyboard navigation and screen reader support Performance ■ Lazy Loading : Load sub-menu content on demand Efficient Rendering : Use OnPush change detection strategy Memory Management : Clean up event listeners and subscriptions Styling ■ The sidebar component uses CSS custom properties for theming: CSS Custom Properties ■ Property Default Value Description --sidebar-background #ffffff Background color of the sidebar --sidebar-border #e2e8f0 Border color of the sidebar

```
<div class="sidebar-demo">
  <aava-sidebar
    [width]="sidebarWidth()"
    [collapsedWidth]="collapsedSidebarWidth()"
    [showCollapseButton]="true"
    [buttonVariant]="'inside'"
    [isCollapsed]="isCollapsed()"
    [position]="'left'"
    (collapseToggle)="onCollapseToggle($event)"
  >
    <div class="demo-header-content">
      <span class="header-title">My App</span>
    </div>

    <ul slot="content" class="demo-content">
      <li
        *ngFor="let item of menuItems"
        [class.active]="item.active"
        class="nav-item"
      >
        <span class="nav-icon">{{ item.icon }}</span>
        <span *ngIf="!isCollapsed()">{{ item.label }}</span>
      </li>
    </ul>
  </aava-sidebar>
</div>


---

isCollapsed = signal<boolean>(false);
sidebarWidth = signal<string>('280px');
collapsedSidebarWidth = signal<string>('70px');

menuItems = [
  { icon: '■', label: 'Dashboard', active: true },
  { icon: '■', label: 'Users' },
  { icon: '■', label: 'Projects' },
  { icon: '■', label: 'Analytics' },
  { icon: '■■', label: 'Settings' },
];

onCollapseToggle(collapsed: boolean): void {
  this.isCollapsed.set(collapsed);
}
```